AAS 16-527

REAL-TIME OPTIMAL CONTROL AND TARGET ASSIGNMENT FOR AUTONOMOUS IN-ORBIT SATELLITE ASSEMBLY FROM A MODULAR HETEROGENEOUS SWARM

Rebecca C. Foust [*], Soon-Jo Chung [†], and Fred Y. Hadaegh[‡]

This paper presents a decentralized optimal guidance and control scheme to combine a heterogeneous swarm of component satellites, rods and connectors, into a large satellite structure. By expanding prior work on a decentralized auction algorithm with model predictive control using sequential convex programming (MPC-SCP) to allow for the limited type heterogeneity and docking ability required for in-orbit assembly. The assignment is performed using a distributed auction with a variable number of targets and strict bonding rules to address the heterogeneity. MPC-SCP is used to generate the collision-free trajectories, with modifications to the constraints to allow docking.

INTRODUCTION

This paper presents a decentralized guidance and control scheme to combine a heterogeneous swarm of component satellites into a large satellite structure. Design and construction of large space systems is often constrained by factors that have more to do with surviving launch than the intended mission, like launch vehicle fairing size or ability to withstand launch loading. Satellites constructed in space would not experience these design constraints, allowing for lighter, more capable satellites. Start to finish construction in orbit is not yet possible, but improvements can still be made through recent advances in swarm spacecraft guidance and control[1,2,3] and autonomous rendezvous and docking.[4] With two types of satellites, a rod and a multiport connector, a wide variety of shapes can be created to support many different missions. The rod is a rectangle with two docking ports located on the ends. The connector satellite is a regular hexagon with six docking ports along the sides. The advantages of such a mission are clear: increased reliability due to redundancy, increased flexibility in final configuration, and ability to reconfigure for future missions. The mission concept is illustrated in Figure 1. The steps are as follows:

Step 1 The components enter into loose, collision-free J_2 invariant passive relative orbits.[1]

Step 2 The components determine their desired final position in the assembly and move to take the position using a modified version of SATO.

Step 3 Along the path to the final position, components assigned to neighboring positions dock and proceed combined.

[*]Graduate Student, Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA.
[†]Associate Professor, Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA.
[‡]Senior Research Scientist and Technical Fellow, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109, USA.

Step 4 Finally, a complete structure is made once all components have reached their final destination.

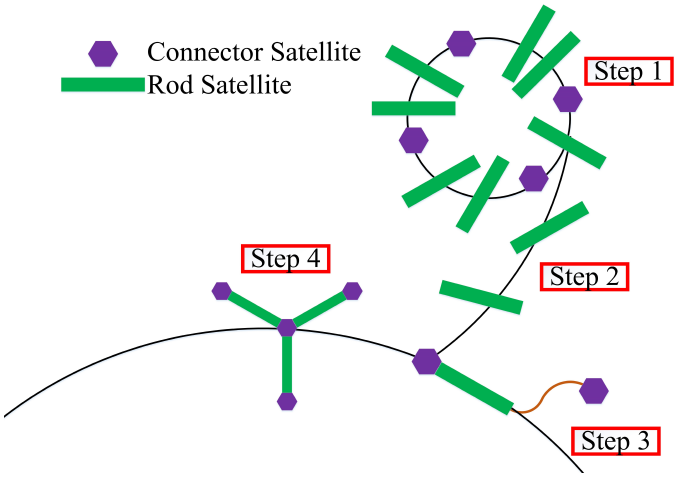


Figure 1: Outline of Mission Steps

A similar modular swarm construction mission was demonstrated using a homogeneous swarm of rectangular boats constructed in a brick pattern.[5] Though this demonstration involved a homogeneous swarm with a planar construction and centralized guidance and assignment, the assembly scheme is multi-seeded similar to the present paper.

Many examples of decentralized swarm guidance schemes exist, but the swarms are typically homogeneous.[6,7,8,9] The heterogeneous swarm guidance schemes typically use centralized algorithms. One heterogeneous swarm used a hierarchical decentralized scheme for coordination of land and air vehicles, but the agents were coordinated centrally among the different types.[10]

This paper intends to create a decentralized assignment and guidance scheme for a heterogeneous swarm. To achieve this, prior work in homogeneous swarms will be leveraged. In prior work, the Swarm Assignment and Trajectory Optimization (SATO) algorithm was used to solve a target assignment and collision-free path planning problem by implementing a decentralized auction algorithm with model predictive control using sequential convex programming (MPC-SCP).[2,11] In addition to the heterogeneity logic, the collision avoidance logic in MPC-SCP must be carefully relaxed to allow docking agents to come within the collision avoidance radius.

OVERVIEW OF SATO ALGORITHM MODIFICATIONS

The Swarm Assignment and Trajectory Optimization (SATO) algorithm is used to solve a target assignment and collision-free path planning problem by implementing a decentralized auction algorithm with optimal trajectory generation.[11] The two main parts of this algorithm are the target assignment auction, Variable Swarm Distributed Auction Algorithm (VSDAA), and trajectory generation, Model Predictive Control using Sequential Convex Programming (MPC-SCP). The two algorithms are run sequentially over the course of the SATO algorithm so that the initial assignments and trajectories can be updated as agent connectivity changes or collision avoidance is needed. All agents are assumed to know of the set of target locations, and have a limited communication radius.

This algorithm performs well for reassigning homogeneous swarms, but needs modification to

handle the heterogeneity and to incorporate the docking schemes described above. The changes made will be outlined explicitly in the following sections.

Variable-Swarm Distributed Auction Algorithm (VSDAA) with Modifications

The nominal VSDAA algorithm assumes each agent in the swarm knows the location of all the possible targets. Since the algorithm is decentralized, all bidding information is communicated only to agents within a communication radius. Each agent calculates its cost to each target by using sequential convex programming to solve Problem 1, a trajectory generation problem ignoring collision avoidance, or using the distance between the two points. The cost function used here and in the trajectory generation section was chosen for to create spacecraft fuel optimal paths, though the vector norm chosen depends the spacecraft thruster configuration.[2] The agent then uses this cost to bid for target locations with the agents within its communication radius. The auction is performed for twice the diameter of the communication network, allowing all bids to propagate completely through the communicating agents. As the agents move to the targets, the communication graph becomes connected which ensures that over time the near-optimal assignment will be reached.[11]

Problem 1 (Auction Cost)**.**

$$\min_{\mathbf{u}_j} \sum_{k=k_0}^{T-1} \|\mathbf{u}_j[k]\|_1 \Delta t \quad \text{subject to} \tag{1}$$

$$\mathbf{x}_j[k+1] = A_j[k]\mathbf{x}_j[k] + B_j[k]\mathbf{u}_j[k] + z_j[k], \qquad k = k_0, \ldots, T-1, \qquad j = 1, \ldots, N \tag{2}$$

$$\|\mathbf{u}_j[k]\|_\infty \leq U_{\max} \qquad k = k_0, \ldots, T-1, \qquad j = 1, \ldots, N \tag{3}$$

$$\|H\mathbf{x}_j[k]\|_2 \leq V_{\max} \qquad H = [0_{3\times 3} \quad I_{3\times 3}], \qquad k = k_0, \ldots, T, \qquad j = 1, \ldots, N \tag{4}$$

$$\mathbf{x}_j[0] = \mathbf{x}_{j,0} \tag{5}$$

$$\mathbf{x}_j[T] = \mathcal{X}_f(j) \tag{6}$$

VSDAA has to be modified to ensure that each agent type is assigned to an appropriate location. For the current agent definitions, the agent types have different number and location of docking ports but the same radius, as seen in Figure 2. This means that potential target locations can be differentiated by number and the angle between docks required at each location. Algorithmically, this involves changing the cost function used in the auction to make improper assignments prohibitively expensive. This is achieved through the use of barrier functions. The target information known by every agent must now indicate the location of the target and how many docks must be performed at that location. The modified assignment problem is presented in Problem 2.

Problem 2 (Assignment Problem)**.**

$$\min_{\mathbf{x}_{j,f},\, j=1\ldots N} \sum_{j=1}^{N} [C(\mathbf{x}_{j,0}, \mathbf{x}_{j,f}) + \mathcal{B}(n_j, N_T(\mathbf{x}_{j,f})] \quad \text{subject to} \tag{7}$$

$$\mathbf{x}_{j,f} \in \mathcal{X}_f, \qquad \mathbf{x}_{j,f} \neq \mathbf{x}_{i,f}, \qquad \forall j = 1 \ldots N, \qquad \forall i \neq j$$

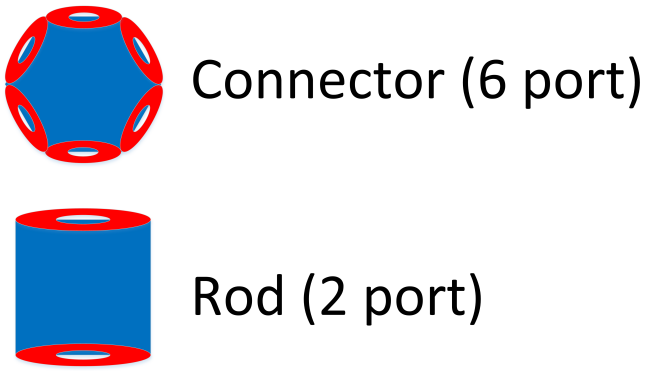


Figure 2: Definition of Rod and Connector Agent Types

where $C(\mathbf{x}_0, \mathbf{x}_f)$ is the cost required for an agent to go from $\mathbf{x}_0$ to $\mathbf{x}_f$ (the solution to Problem 1) and $\mathcal{B}(n, N_T(\mathbf{x}_f))$ is the barrier function to prevent agents from targeting locations they cannot accommodate. $N_T(\mathbf{x}_f)$ is the number of docks required at a target and n is the maximum number of docks an agent can perform based on its type (6 for connectors, 2 for rods).

An example barrier function is:

$$\mathcal{B}(n, N_T(\mathbf{x}_f)) = -\log(g(n, N_T(\mathbf{x}_f)) + 1) \tag{8}$$

for some sigmoid $g(n, N_T(\mathbf{x}_f))$ like

$$g(n, N_T(\mathbf{x}_f)) = \begin{cases} n - N_T(\mathbf{x}_f) & n \geq N_T(\mathbf{x}_f) \\ -1 & n < N_T(\mathbf{x}_f) \end{cases} \tag{9}$$

chosen to give $\mathcal{B}(n, N_T(\mathbf{x}_f))$ infinite value when the number of docks at a target exceeds the number of docks the agent can perform.
It is also possible to use a different sigmoid function to change the performance of the assignment by changing the function $a(n, N_T(\mathbf{x}_f))$.

$$B(n, N_T(\mathbf{x}_f)) = \begin{cases} -log(a(n, N_T(\mathbf{x}_f)) & n \geq N_T(\mathbf{x}_f) \\ Inf & n < N_T(\mathbf{x}_f) \end{cases}$$

Three different $a(n, N_T(\mathbf{x}_f))$ are shown in figure 3. Using $a(n, N_T(\mathbf{x}_f)) = 1$ gives a simple sorting of agents. Changing $a(n, N_T(\mathbf{x}_f))$ to some decreasing positive function of $n - N_T(\mathbf{x}_f)$ like $1/(n - N_T(\mathbf{x}_f) + 1)$ makes agents inclined towards positions where they are most useful. The chosen barrier function uses $a(n, N_T(\mathbf{x}_f)) = n - N_T(\mathbf{x}_f)$ to dissuade agents from using all of their docking ports. If no barrier function is used then the problem is the same as the standard VSDAA. The algorithm implementing the solution to this problem is presented in Method 1.

The above barrier functions would not be sufficient to sort out all cases, like the one illustrated in Figure 4. In assignments with underutilized connectors, with one or two docks, it is necessary to augment the barrier function to include the angle between the docks so that improper assignments are avoided. For now, we will assume the assignments avoid underutilized connectors.

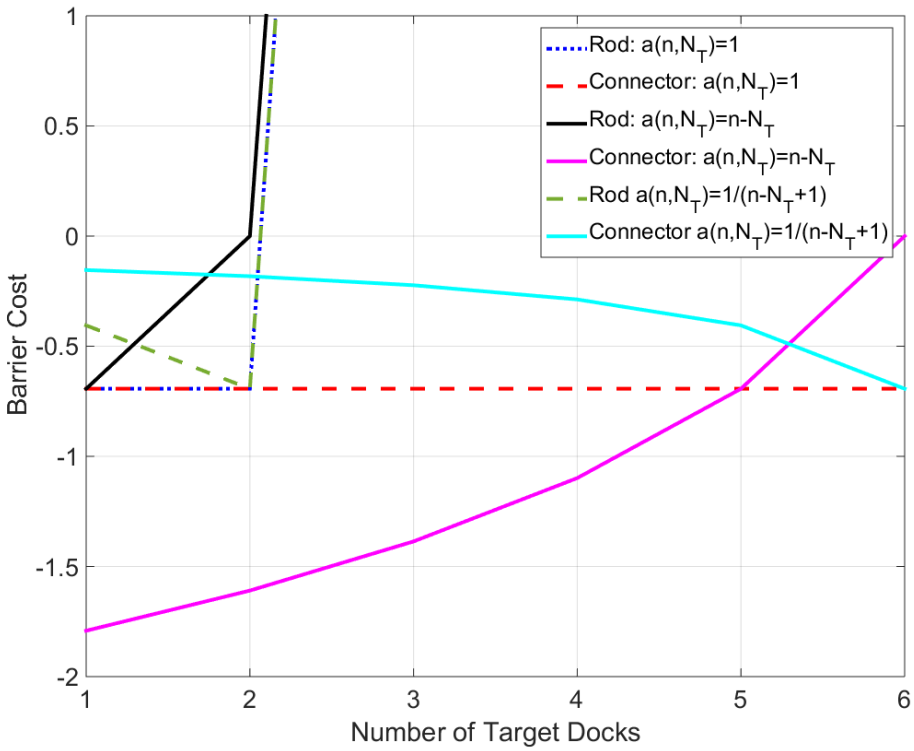


Figure 3: Possible barrier functions to use in assignment

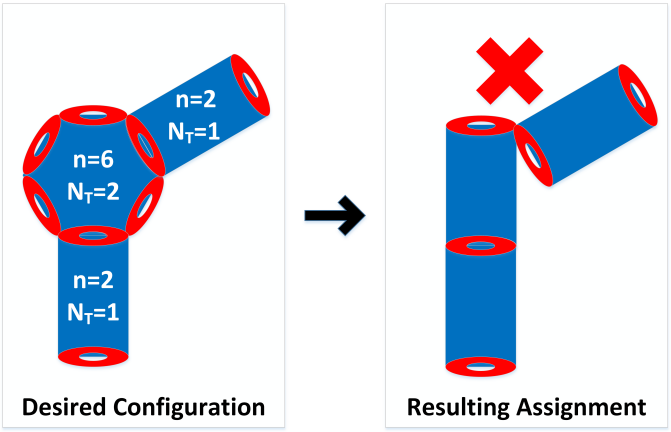


Figure 4: Bad Assignment: the desired configuration on the left has an underutilized connector, with only two docks required. This allows the initial barrier function to mis-assign a rod agent to the connector location, resulting in a disconnected structure.

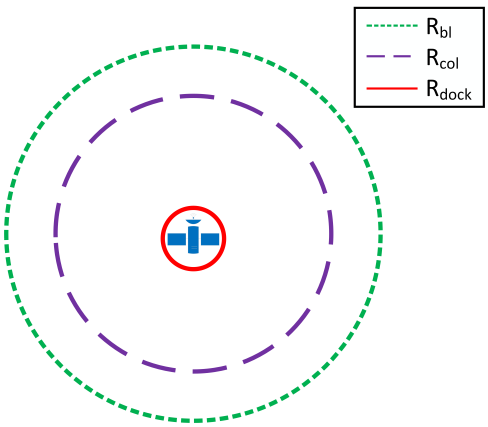
Method 1 Variable-Swarm, Distributed Auction Algorithm for Docking (VSDAA-D)[11]

1: $\mathcal{X}_f$ = terminal positions in desired shape
2: $\mathbf{c}^i(s)$ = cost of agent i choosing target s
3: $\mathbf{b}^i(s)$ = docking barrier function for agent i choosing target s
4: m^i = # of targets available for agent i to bid on
5: $\mathbf{p}^i = \mathbf{0}_{1\times m^i}$
6: $\mathbf{p}^i_{\text{old}} = -\mathbf{1}_{1\times m^i}$
7: $j^i = 1$
8: $count^i = 0$
9: **for all** i (*run in parallel*) **do**
10: **while** $count^i < 2D_{\text{net}}$ **do**
11: **if** $|\mathbf{p}^i(j^i)| > \mathbf{p}^i_{\text{old}}(j^i)$ (i *is outbid*) **then**
12: $m^i = \max\left(m^i, |\{s|\mathbf{p}^i(s) \neq 0\}|\right)$
13: **if** $|\{s|\mathbf{p}^i(s) > 0\}| = m^i$ **then**
14: $m^i = |\{s|\mathbf{p}^i(s) > 0\}| + 1$
15: $\mathbf{p}^i(1:m^i) = -\left(|\mathbf{p}^i(1:m^i)| + \epsilon\right)$
16: **end if**
17: $v^i = \min_{s=1\ldots m^i}\left(\mathbf{c}^i(s) + \mathbf{b}^i(s) + |\mathbf{p}^i(s)|\right)$
18: $j^i = \arg\min_{s=1\ldots m^i}\left(\mathbf{c}^i(s) + \mathbf{b}^i(s) + |\mathbf{p}^i(s)|\right)$
19: $w^i = \min_{s=1\ldots m^i, s\neq j^i}\left(\mathbf{c}^i(s) + \mathbf{b}^i(s) + |\mathbf{p}^i(s)|\right)$
20: $\gamma^i = w^i - v^i + \epsilon$
21: $\mathbf{p}^i(j^i) = |\mathbf{p}^i(j^i)| + \gamma^i$
22: $count^i = 0$
23: **else if** $\mathbf{p}^i \neq \mathbf{p}^i_{\text{old}}$ (*another agent is outbid*) **then**
24: $m^i = \max\left(m^i, |\{s|\mathbf{p}^i(s) \neq 0\}|\right)$
25: $count^i = 0$
26: **else**
27: $count^i = count^i + 1$
28: **end if**
29: $\mathbf{p}^i_{\text{old}} = \mathbf{p}^i$
30: Communicate $\mathbf{p}^i$ to all agents in $\mathcal{N}_{[i]}$
31: **for** $s = 1\ldots m^i$ **do**
32: $\mathbf{p}^i(s) = \min_{q\in\arg\max_{q\in\mathcal{N}_{[i]}}(|\mathbf{p}^q(s)|)}\left(\mathbf{p}^q(s)\right)$
33: **end for**
34: **end while**
35: *Optional:* $m^i = |\{j|\mathbf{p}^i(j) \neq 0\}|$
36: *Optional:* Go back to line 5 and rerun with new m^i
37: $\mathbf{x}_{i,f} = \mathcal{X}_f(j^i)$
38: **end for**

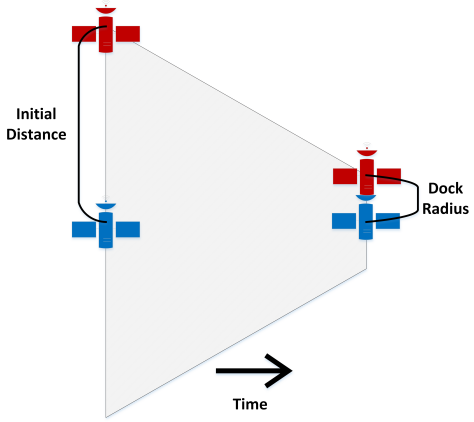
Trajectory Generation

The next modification was to the trajectory generation portion of the SATO algorithm. MPC-SCP is used to create the optimal, collision-free trajectories to the targets selected by VSDAA. Initially, a nominal trajectory is generated without considering collision avoidance. Then each agent solves the MPC-SCP problem with collision avoidance on a limited time horizon, with the knowledge of the nominal trajectories of the agents within its communication radius. The collision avoidance constraint is not convex, but by approximating the other agents' collision avoidance spheres as hyperplanes orthogonal to the surface, convexity can be obtained.

Because the agents need to dock for construction, the collision avoidance constraint must be suspended for agents that are attempting to dock. This is achieved using a boundary layer around the collision avoidance radius. The relative sizes of the radii are illustrated in Figure 5a. When an agent approaches within the boundary layer, the main agent checks to see if that agent is targeted for a location neighboring the main agent's target. If it is, the agent follows the docking constraint. Otherwise, the collision avoidance constraints hold.



(a) Relative sizes of the radii crucial to the modified trajectory generation algorithm. When another agent comes within the boundary layer radius, the main agent decides whether to dock or avoid it.



(b) Description of docking cone used in modified MPC-SCP. The initial distance is the distance when the docking constraint is first applied, when the red agent comes within the boundary layer of the blue agent or vice versa.

Figure 5: Explanation of docking radii and cone

In order to allow the agents some flexibility in docking, the docking constraint requires the agent to maintain a shrinking distance to the agent to be docked. This means that over time the main agent will stay within a cone defined by the other agent. The cone radius begins as the initial separation and ends as the agent docking radius, as seen in Figure 5b. This docking cone forces the agents to come together by the final time. This way, the agents are allowed to dock before they reach the target if it is beneficial to their trajectories, or they can wait until the final position to dock.

The modified trajectory generation problem to be solved is given in Problem 3.

Problem 3 (Trajectory Generation)**.**

$$\min_{\mathbf{u}_j} \sum_{k=k_0}^{T-1} \|\mathbf{u}_j[k]\|_1 \Delta t \tag{10}$$

subject to the following constraints:

(a) the dynamics, state, and control constraints

$$\mathbf{x}_j[k+1] = A_j[k]\mathbf{x}_j[k] + B_j[k]\mathbf{u}_j[k] + z_j[k], \qquad k = k_0, \ldots, T-1, \qquad j = 1, \ldots, N \tag{11}$$

$$\|\mathbf{u}_j[k]\|_\infty \leq U_{\max} \qquad k = k_0, \ldots, T-1, \qquad j = 1, \ldots, N \tag{12}$$

$$\|H\mathbf{x}_j[k]\|_2 \leq V_{\max} \qquad H = [0_{3\times 3} \quad I_{3\times 3}], \qquad k = k_0, \ldots, T, \qquad j = 1, \ldots, N \tag{13}$$

(b) the initial and terminal conditions obtained from Problem 2

$$\mathbf{x}_j[0] = \mathbf{x}_{j,0}, \qquad \mathbf{x}_j[T] = \mathbf{x}_{j,f}, \qquad j = 1, \ldots, N \tag{14}$$

(c) the convexified constraint of collision avoidance[2]

$$\begin{aligned} &(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k])^T G^T G(\mathbf{x}_j[k] - \bar{\mathbf{x}}_i[k]) \geq R_{\text{col}} \|G(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k])\|_2 \\ G = [I_{3\times 3} \quad 0_{3\times 3}], &\qquad k = k_{bl}, \ldots, \min\{k_0 + T_H, T\}, \qquad i \in \mathcal{N}_{[j]} \cap \mathcal{P}_j \setminus \mathcal{D}_j \end{aligned} \tag{15}$$

where $\mathcal{N}_{[j]} = \{i |\ \|\mathbf{x}_j[k_0] - \mathbf{x}_i[k_0]\|_2 \leq R_{\text{comm}}\}$ and R_{comm} is the communication radius of each agent. Also, $\mathcal{D}_j$ is the set of agents that are assigned to dock with agent j and $\mathcal{P}_j$ is the set of agents that have a higher priority than j (see Morgan et al.[11]).

(d) the new docking condition
if $\|G(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k]\|_2 \leq R_{\text{bl}}$:

$$\|G(\mathbf{x}_j[k] - \bar{\mathbf{x}}_i[k])\|_2 \leq R_{\text{cone}}(k), \quad k_{bl} = \arg\min_{k_0 \leq k \leq k_0 + T_H} \{\|G(\bar{\mathbf{x}}_j[k] - \bar{\mathbf{x}}_i[k]\|_2 - R_{\text{bl}}\} \tag{16}$$

$$\begin{aligned} &R_{\text{cone}}(k) = R_{\text{dock}} + \frac{\|G(\mathbf{x}_j[k_{bl}] - \bar{\mathbf{x}}_i[k_{bl}])\|_2 - R_{\text{dock}}}{T - k_{bl}}(T - k), \\ &k = k_{bl}, \ldots, \min\{k_0 + T_H, T\},\ i \in \mathcal{N}_{[j]} \cap \mathcal{P}_j \cap \mathcal{D}_j \end{aligned}$$

Problem 3 is solved using the sequential convex programming algorithm in Method 2. The addition of attitude dynamics changes the dimensions of the above variables but does not require any additional logic.

Method 2 Sequential Convex Programming [2]

$\bar{\mathbf{x}}_j[k] := \mathbf{0}_{6\times 1},\ \forall j,k$
$\mathbf{x}_{j,0}[k] :=$ the solution to Problem 3 (Trajectory Generation) with $\mathcal{P}_j = \emptyset,\ \forall j,k$
$\bar{\mathbf{x}}_j[k] := \mathbf{x}_j^0[k],\ \forall j,k$
Communicate $\bar{\mathbf{x}}_j[k]$ to all neighboring agents ($i \in \mathcal{N}_{[j]}$)
$\mathcal{K} := \{1,\dots,N\}$
$w := 1$
while $\mathcal{K} \neq \emptyset$ **do**
  for all $j \in \mathcal{K}$ (*run in parallel*) **do**
    $\mathbf{x}_{j,w}[k] :=$the solution to Problem 3 (Trajectory Generation), $\forall k$
  end for
  for all j (*run in parallel*) **do**
    $\bar{\mathbf{x}}_j[k] := \mathbf{x}_{j,w}[k],\ \forall k$
    Communicate $\bar{\mathbf{x}}_j[k]$ to all neighboring agents ($i \in \mathcal{N}_{[j]}$)
    if $\|\mathbf{x}_{j,w}[k] - \mathbf{x}_{j,w-1}[k]\|_\infty < \epsilon_{\text{SCP}}\ \forall k$ and $\|G(\mathbf{x}_{j,w}[k] - \mathbf{x}_{i,w}[k])\|_2 > R_{\text{col}}\ \forall k \geq k_{bl}, \forall i \in \mathcal{N}_{[j]} \cap \mathcal{P}_j \setminus \mathcal{D}_j$ **then**
      Remove j from $\mathcal{K}$
    end if
  end for
  $w := w + 1$
end while

A model predictive control implementation of SCP (Method 2) can be used to implement the VSDAA and SCP methods in real time in order to simultaneously solve the swarm assignment and trajectory optimization (SATO) problems. MPC uses a receding horizon to update the optimal target assignments for docking (Method 1) and current trajectories obtained via communication with neighbors and on-board sensors. SATO is described in Method 3.

SIMULATION RESULTS

The modified SATO algorithm was implemented in a MATLAB simulation with three different dynamical environments, two-dimensional double integrator dynamics with and without attitude and three-dimensional spacecraft dynamics without attitude dynamics.

Simple Dynamics

The first dynamics used were two dimensional double integrator dynamics, $\ddot{x} = 1$. The agents are targeted to make a seven-hexagon flower using 24 connectors and 42 rods, with each connector agent docking with three rod agents to prevent improper assignments. The full simulation, shown in figure 6 is too dense, so the majority of deviations from the initial trajectory are for collision avoidance. To better illustrate the docking paths, the results shown below are from a 6 connector, 12 rod simulation. Figure 7 shows the trajectories resulting from the simulation. Connector agents have hexagon markers at the initial positions and rod agents have circle markers. The target locations are blue squares.

Collision avoidance can be seen in several places in Figure 7 where trajectories make sharp movements. The docking constraint is in effect for trajectories that should be avoiding collision, but are

Method 3 Swarm Assignment and Trajectory Optimization (SATO) [11]

$k_0 = 0$
while $k_0 \leq T$ **do**
  for all $i = 1, \ldots, N$ (*parallel*) **do**
    for all $j = 1, \ldots, M$ **do**
      Solve Problem 1 using SCP (Method 2)
      $\mathbf{c}^i(j) =$ cost of optimal solution to Problem 1
    end for
  end for
  Solve Problem 2 using VSDAA (Method 1)
  $\mathbf{x}_{j,f} =$ solution to Problem 2, $\forall j$
  if # of bids has changed **then**
    $k_0 = 0$
  end if
  Solve Problem 3 using SCP (Method 2)
  $\mathbf{u}_j[k] =$ control solution to Problem 3, $\forall j, k = k_0 \ldots k_0 + T_H - 1$
  Apply $\mathbf{u}_j[k]$ for $k = k_0 \ldots k_0 + T_H - 1$
  Update k_0 and $\mathbf{x}_{j,k_0}$ to current time
end while

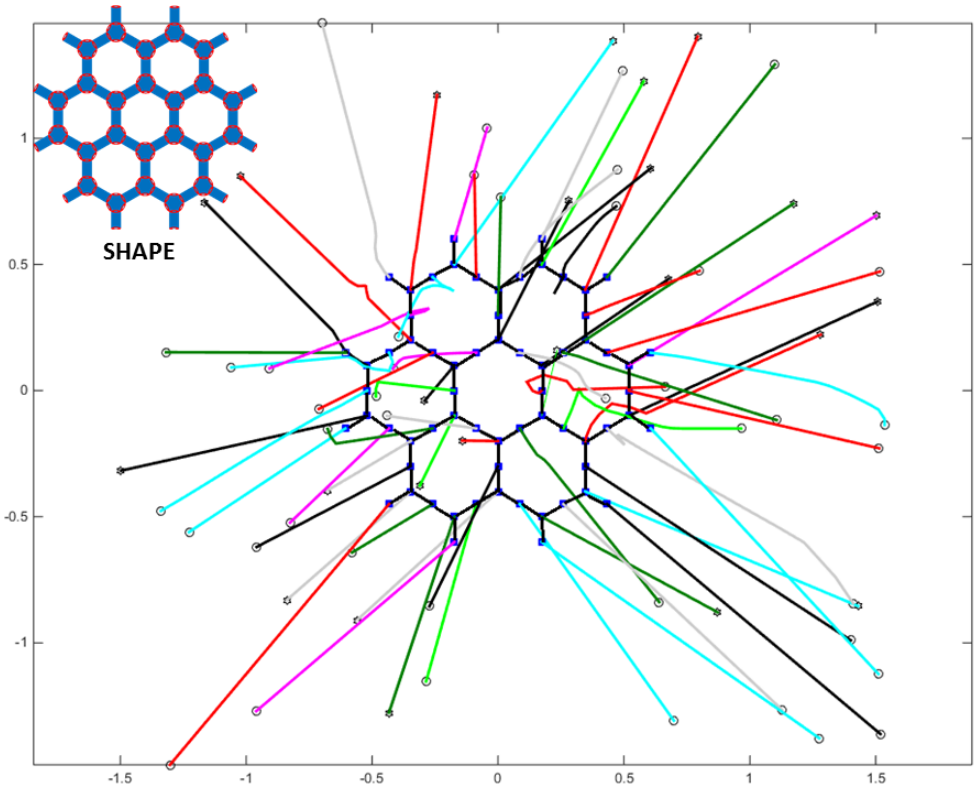


Figure 6: 66 Agent simulation of SATO-H with simple dynamics. Hexagon agents are connectors and circle agents are rods. The blue squares making the flower shape in the center are the target locations.

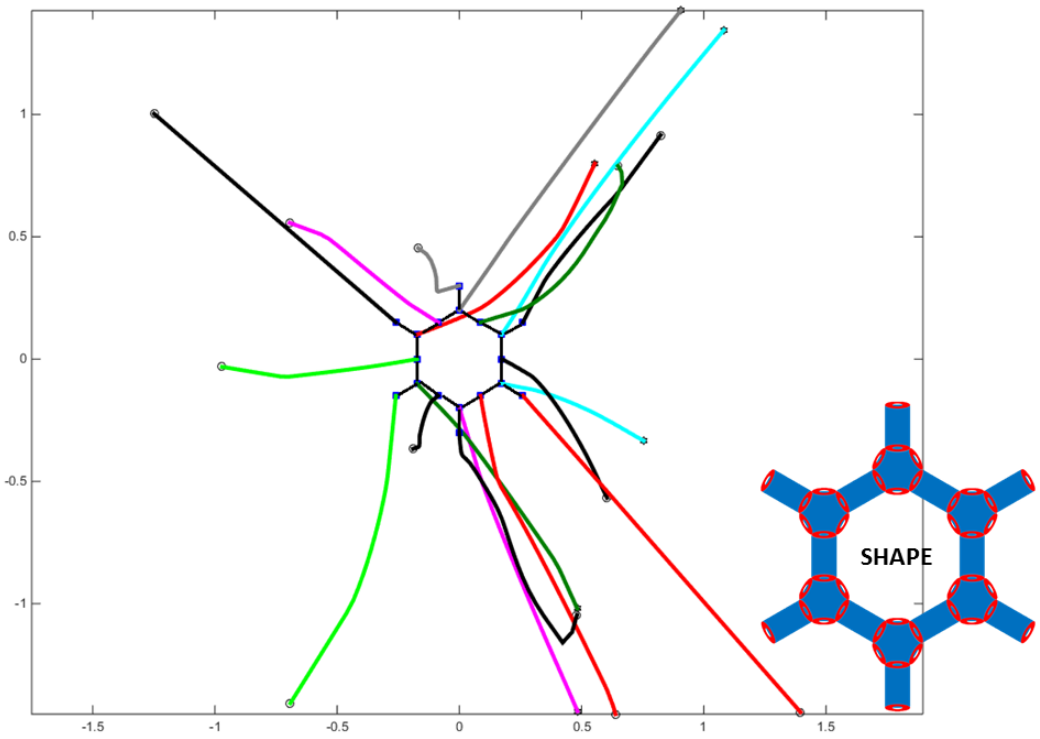


Figure 7: 18 Agent simulation of SATO-H with simple dynamics. Hexagon agents are connectors and circle agents are rods. The blue squares making the flower shape in the center are the target locations.

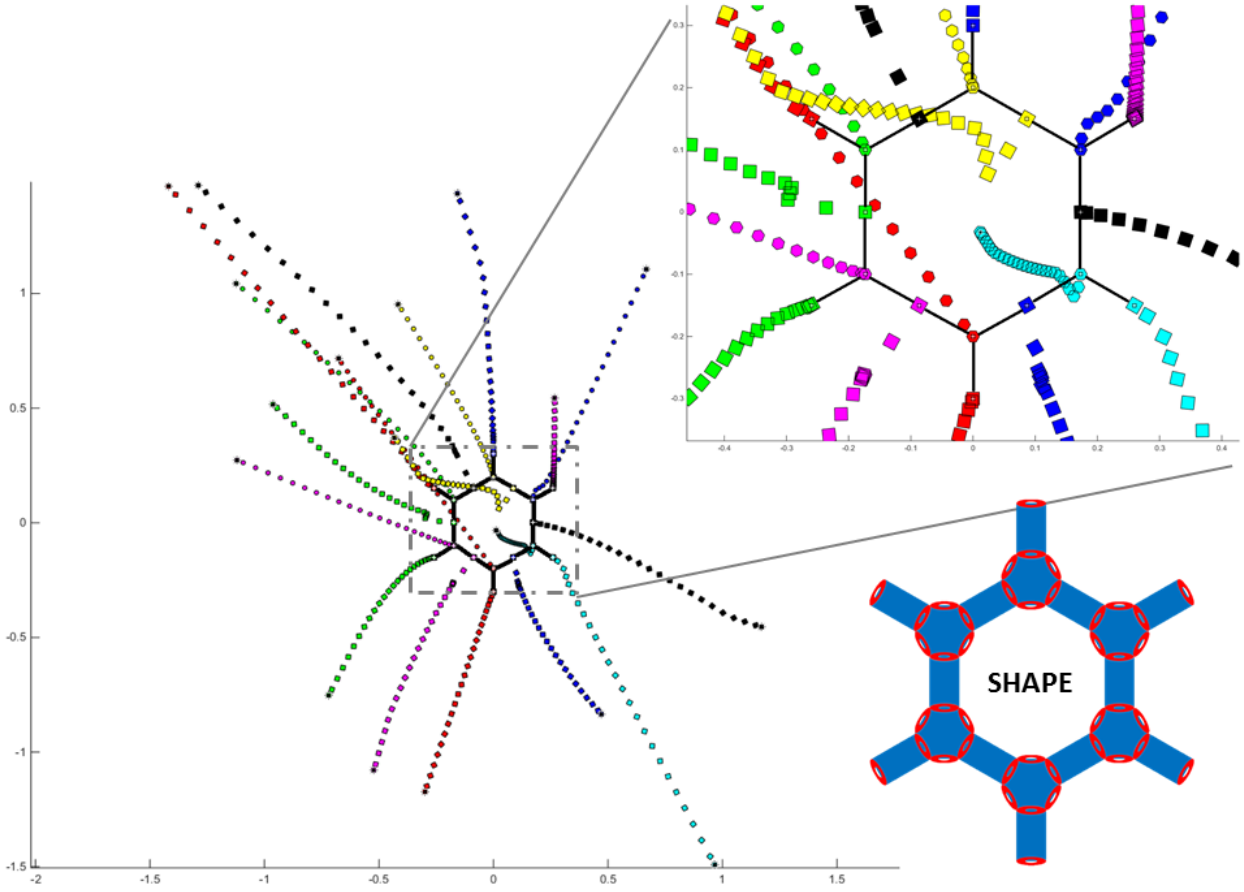


Figure 8: 18 Agent simulation of SATO-H with simple dynamics plus attitude modeling. Hexagon agents are connectors and square agents are rods. The angle of the symbol represents the attitude of the agent.

not. All terminal positions are within the collision avoidance distance, so each agent is using the docking constraint. One issue with the current formulation is that each agent avoids everything but its immediate neighbors, regardless of whether an agent it is avoiding is directly connected to an immediate neighbor. This will be resolved in future configurations.

Simple Dynamics with Attitude

In this example, the state vector is expanded to include orientation, θ. The angle is also governed by double integrator dynamics, $\ddot{\theta} = 1$. Including the orientation of the agents requires some additional thought with regard to terminal orientation constraints. Currently, the final orientation is given with the set of targets, but this can be suboptimal since the agents could exploit symmetry in docking port locations to minimize fuel use. In future work the agents will choose their orientation relative to docking agents to minimize fuel costs while satisfying a docking angle constraint. The constraint will need to be upheld when the agents enter docking proximity instead of at the final time. This will tie the orientation and docking constraints so that where ever the agent docks inside the docking cone, the orientation is actually one which will allow docking. The results of the current simulation are presented in Figures 8. The markers are angled to represent the agent orientation at

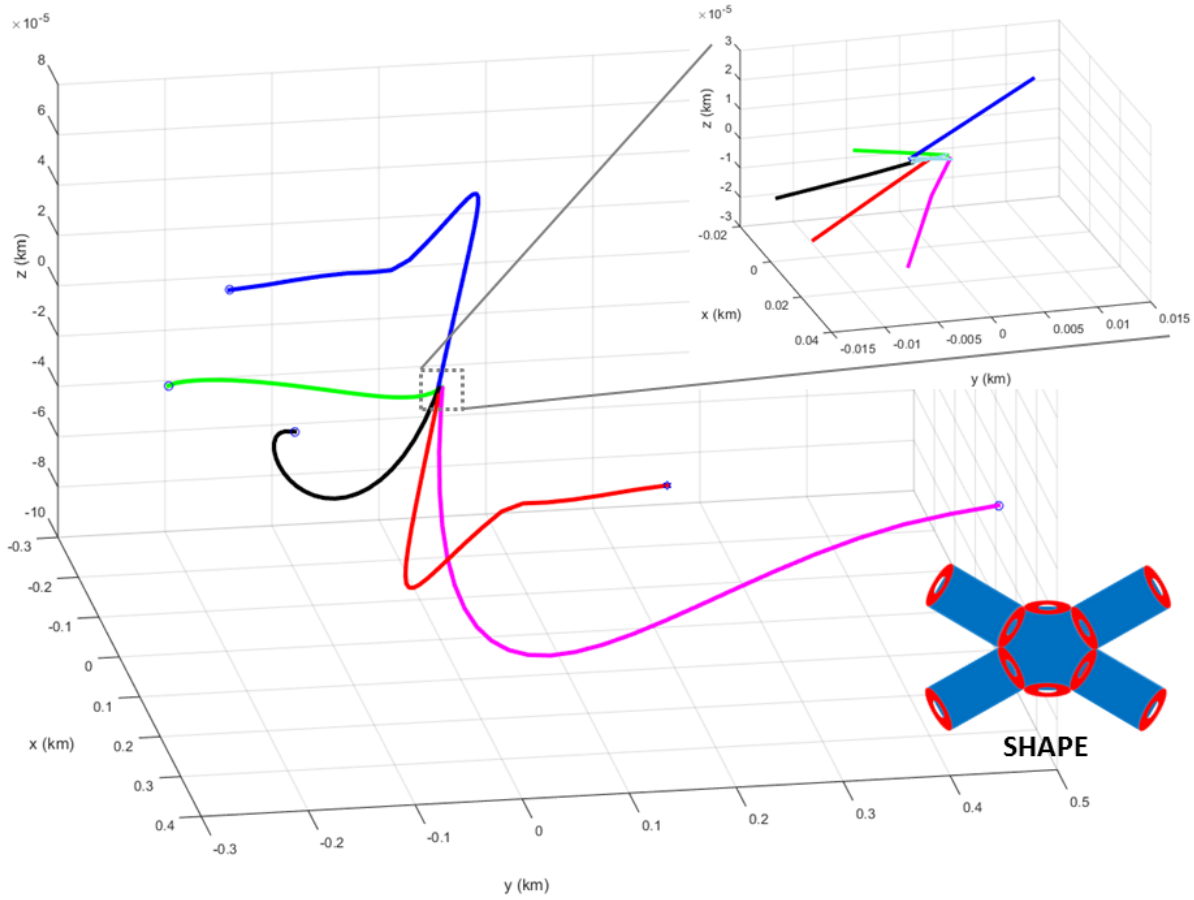


Figure 9: 5 Agent simulation of SATO-H with spacecraft dynamics. Hexagon agents are connectors and circle agents are rods. The final positions have a separation of 2 m.

each step. The black and white squares show the target positions. The docking constraint can be seen in action in the upper center blue square and yellow hexagon. The two trajectories come much closer than the collision avoidance constraint would allow.

3D Spacecraft Dynamics

This example implements the above algorithms using high-fidelity relative orbit dynamics[1] with J_2 perturbations with a virtual chief in a 500 km, 45° inclination orbit. The simulation results in Figures 9 use one connector and four rods targeted to a planar cross with zero relative velocity, though the trajectories to achieve the arrangement are three dimensional. The code uses J_2 invariant relative orbits for the initial positions, which greatly reduce the energy required to maintain the orbit and would likely be used for agents awaiting docking. The agents have an initial separation of up to three kilometers and final separation of two meters.

CONCLUSION

A distributed algorithm has been presented to allow for construction using a heterogeneous swarm of component satellites with limited communication radii. This extends prior work in the field because it is both distributed and heterogeneous, and can function in a complex dynamic environment. Some work remains to address issues like docking with an already docked agent, reducing orientation change by allowing a set of terminal orientations, and enforcing dock distance once achieved. Additionally, the spacecraft dynamics simulation will be extended to include spacecraft attitude dynamics. The algorithm will also be tested on a quadrotor swarm testbed.

ACKNOWLEDGMENTS

This work was supported by a NASA Office of the Chief Technologist Space Technology Research Fellowship. Government sponsorship is acknowledged. This research was carried out in part at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the NASA.

NOTATION

N	number of spacecraft
N_j	number of docks required at a target
n_i	maximum number of docks a satellite can perform based on its type
t	time
t_f	final time ($t_f = T\Delta t$)
t_{run}	time required to compute the optimization
Δt	length of time step
$\mathcal{N}_{[j]}$	set of the closed neighborhood
$\mathcal{D}_j$	set of agents that are assigned to dock with agent j
$\mathcal{P}_j$	set of agents that have a higher priority than j
R_{col}	minimum distance between spacecraft to avoid a collision in the optimization
R_{bl}	distance at which the agent chooses to dock or avoid an approaching agent
R_{dock}	physical separation of centroids of docking agents
$\bar{R}_{\text{col}}$	minimum distance between spacecraft to avoid a collision in reality
R_{comm}	maximum distance a spacecraft can communicate ($R_{\text{comm}} > R_{\text{col}}$)
T	total number of time steps
T_H	number of time steps in the model predictive control horizon
$U_{\max}$	maximum allowable magnitude of the control vector
$V_{\max}$	maximum allowable magnitude of the relative velocity vector
$h_j(\mathbf{x}[k], k])$	cost to transfer a spacecraft from $\mathbf{x}[k]$ at time k to $\mathbf{x}_f$ at T
k	time step k
k_0	time step at the start of the model predictive control horizon
$\mathbf{x}_0$	state vector at initial time
$\mathbf{x}_f$	state vector at final time
$\mathbf{x}_j$	state vector of spacecraft j
$\bar{\mathbf{x}}$	nominal state vector
$\|\cdot\|$	2-norm of a vector

REFERENCES

[1] D. Morgan, S.-J. Chung, L. Blackmore, B. Acikmese, D. Bayard, and F. Y. Hadaegh, "Swarm-keeping strategies for spacecraft under J2 and atmospheric drag perturbations," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 5, 2012, pp. 1492–1506.

[2] D. Morgan, S.-J. Chung, and F. Y. Hadaegh, "Model predictive control of swarms of spacecraft using sequential convex programming," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 6, 2014, pp. 1725–1740.

[3] F. Y. Hadaegh, S.-J. Chung, and H. M. Manaroha, "On Development of 100-gram-class Spacecraft for Swarm Applications," *IEEE Systems Journal*, 2014. in press, DOI:10.1109/JSYST.2014.2327972.

[4] W. Fehse, *Automated rendezvous and docking of spacecraft*, Vol. 16. Cambridge university press, 2003.

[5] J. Seo, M. Yim, and V. Kumar, "Assembly planning for planar structures of a brick wall pattern with rectangular modular robots," *2013 IEEE International Conference on Automation Science and Engineering (CASE)*, IEEE, 2013, pp. 1016–1021.

[6] J. Yu, S.-J. Chung, and P. G. Voulgaris, "Target assignment in robotic networks: Distance optimality guarantees and hierarchical strategies," *IEEE Transactions on Automatic Control*, Vol. 60, No. 2, 2015, pp. 327–341.

[7] M. Turpin, N. Michael, and V. Kumar, "Trajectory planning and assignment in multirobot systems," *Algorithmic Foundations of Robotics X*, pp. 175–190, Springer, 2013.

[8] M. Turpin, K. Mohta, N. Michael, and V. Kumar, "Goal assignment and trajectory planning for large teams of interchangeable robots," *Autonomous Robots*, Vol. 37, No. 4, 2014, pp. 401–415.

[9] D. Panagou, M. Turpin, and V. Kumar, "Decentralized goal assignment and trajectory generation in multi-robot networks: A multiple lyapunov functions approach," *2014 IEEE International Conference onRobotics and Automation (ICRA)*, IEEE, 2014, pp. 6757–6762.

[10] J. Weaver, A. A. Arroyo, G. Dash, and E. M. Schwartz, "Heterogeneous Collaborative Swarms of Autonomous Agents with Varying Capabilities," *Florida Conference on Recent advances in Robotics, Boca Raton, FL*, Citeseer, 2012, pp. 10–11.

[11] D. Morgan, S.-J. Chung, and F. Y. Hadaegh, "Swarm Assignment and Trajectory Optimization Using Variable-Swarm, Distributed Auction Assignment and Model Predictive Control," *The International Journal of Robotics Research*, 2016. in press.